Fast fundamental frequency estimation

Making a statistically efficient estimator computationally efficient

Nielsen, Jesper Kjær; Jensen, Tobias Lindstrøm; Jensen, Jesper Rindom; Christensen, Mads Græsbøll; Jensen, Søren Holdt

Published in:
Signal Processing

DOI (link to publication from Publisher):
10.1016/j.sigpro.2017.01.011

Publication date:
2017

Document Version
Accepted author manuscript, peer reviewed version

Fast Fundamental Frequency Estimation: Making a Statistically Efficient Estimator Computationally Efficient

Jesper Kjær Nielsen[a,b,*], Tobias Lindstrøm Jensen[c,1], Jesper Rindom Jensen[a,2], Mads Græsbøll Christensen[a], Søren Holdt Jensen[c]

[a] *Aalborg University, Audio Analysis Lab, AD:MT, Rendsburggade 14, DK-9000 Aalborg*
[b] *Bang & Olufsen A/S, Acoustic Research, Peter Bangs Vej 15, DK-7600 Struer*
[c] *Aalborg University, Dept. Electronic Systems, Fredrik Bajers Vej 7B, DK-9220 Aalborg*

Abstract

Modelling signals as being periodic is common in many applications. Such periodic signals can be represented by a weighted sum of sinusoids with frequencies being an integer multiple of the fundamental frequency. Due to its widespread use, numerous methods have been proposed to estimate the fundamental frequency, and the maximum likelihood (ML) estimator is the most accurate estimator in statistical terms. When the noise is assumed to be white and Gaussian, the ML estimator is identical to the non-linear least squares (NLS) estimator. Despite being optimal in a statistical sense, the NLS estimator has a high computational complexity. In this paper, we propose an algorithm for lowering this complexity significantly by showing that the NLS estimator can be computed efficiently by solving two Toeplitz-plus-Hankel systems of equations and by exploiting the recursive-in-order matrix structures of these systems. Specifically, the proposed algorithm reduces the time complexity to the same order as that of the popular harmonic summation method which is an approximate NLS estimator. The performance of the proposed algorithm is assessed via Monte Carlo and timing studies. These show that the proposed algorithm speeds up the evaluation of the NLS estimator by a factor of 50-100 for typical scenarios.

Keywords: Fundamental frequency estimation, Toeplitz, Hankel, Fast algorithms, pitch.

1. Introduction

Periodic signals are encountered in many real-world applications such as music processing [1, 2], speech processing [3, 4], sonar [5], order analysis [6], and electrocardiography (ECG) [7]. Such signals can be modelled as a weighted sum of sinusoids whose frequencies are integer multiples of a common fundamental frequency which in audio and speech applications is often referred to as the pitch [2]. Therefore, an important and fundamental problem in the above mentioned applications is to estimate this fundamental frequency from an observed data set. Multiple estimation methods have been proposed in the scientific literature ranging from simple correlation-based methods [8] to parametric methods [2]. Although the parametric methods in general are much more accurate than the correlation-based methods, they suffer from a high computational complexity. Consequently, the correlation-based methods remain very popular despite that they require all sorts of heuristic post-processing to give a satisfactory performance [9–13]. Since many applications require real-time processing, the computational complexity of the parametric methods must be reduced to make them a viable alternative to the correlation-based methods, and the contribution presented in this paper should be seen in this context.

The main difficulty in estimating the fundamental frequency is that a non-linear optimisation problem has to be solved. No closed-form solution is available, and we, therefore, have to search for the global optimiser of an often very oscillatory cost function such as the two examples shown in Fig. 1. This search for the optimiser is often performed using the following steps.

1. The cost function is evaluated on a grid and one or several candidate optimisers are selected on this grid. Often, the grid is uniform since a part of the cost function can then be evaluated efficiently using an FFT algorithm.
2. The candidate optimisers are refined using, e.g., interpolation methods, line searches, or derivative-based methods.

*Corresponding author
Email addresses: jkn@create.aau.dk (Jesper Kjær Nielsen), tlj@es.aau.dk (Tobias Lindstrøm Jensen), jrj@create.aau.dk (Jesper Rindom Jensen), mgc@create.aau.dk (Mads Græsbøll Christensen), shj@es.aau.dk (Søren Holdt Jensen)
[1]The work of T.L. Jensen was supported by the Independent Research Council for Technology and Production 4005-00122.
[2]The work of J.R. Jensen was supported by the Independent Research Council for Technology and Production 1337-00084.

3. For the parametric methods, model order estimation has to be performed when the model is unknown to reduce the risk of estimating an integer multiple or division of the true fundamental frequency. Often, this problem is also referred to as pitch halving/doubling or as octave errors. Estimating an unknown model often means that we have to repeat the first two steps above for every candidate model, thus increasing the computational complexity significantly.

In the correlation-based methods, the cost function is the autocorrelation function (or some variation thereof) which can typically be computed very efficiently. Adding to this, the correlation-based methods are not model based so it is not necessary to do model comparison to determine the number of harmonic components in the signal. From a computational perspective, the correlation-based methods are, therefore, very attractive. Unfortunately, they have a suboptimal estimation performance, are not very robust to noise (see, e.g., Fig. 3), and do not work for low fundamental frequencies [14]. Here, a low frequency means the number of cycles in a segment of data rather than the frequency measured in, e.g., Hz or radians/s, and this also explains the somewhat non-standard value on the x-axis in Fig. 1. The poor performance for low fundamental frequencies is hardly surprising since fewer and fewer data points are used in the computation of the autocorrelation function as the candidate fundamental frequency decreases. As exemplified by the very popular YIN method [11], this is often solved by using data from the previous data segment, but this trick corresponds to doubling the segment length and using a 50% overlap. Thus, the correlation-based methods cannot provide the same time-frequency resolution as those parametric methods which also work for a low fundamental frequency.

The poor noise robustness and time-frequency resolution seem to be fundamental flaws of the correlation-based methods and the main reason for considering alternative estimators based on a parametric model. Unfortunately, the evaluation of the cost function in the parametric methods is often quite numerically costly since they can involve eigenvalue decompositions of covariance matrices [15, 16] or matrix inversions [17, 18]. A notable exception, though, is the harmonic summation method (HS) [19, 20] which is an approximate non-linear least-squares (NLS) estimator and can be implemented efficiently using a single FFT [2]. The HS summation estimator is statistically efficient and

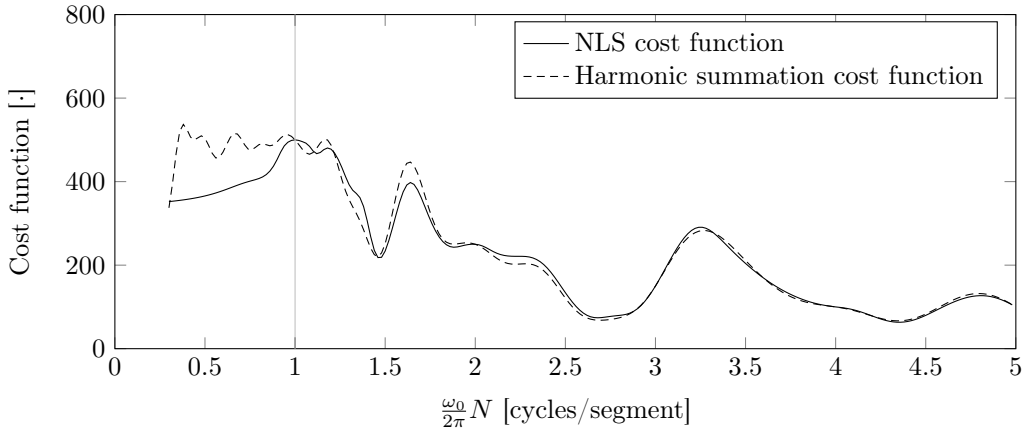

Figure 1: Example of the exact NLS and harmonic summation cost functions. $N = 100$, $L = 10$, $\omega_0 = 2\pi/N$, and constant amplitude $\sqrt{a_i^2 + b_i^2} = 1$, $i = 1, \ldots, L$.

robust to noise and is, therefore, a very attractive alternative to the correlation-based methods. Unfortunately, the HS method also fails for low fundamental frequencies and, therefore, suffers from a suboptimal time-frequency resolution. This is in contrast to the NLS estimator [17] which has a much better performance for low fundamental frequencies and, consequently, a better time-frequency resolution [21], although at a much higher computational complexity. In Fig. 1, the differences and similarities between the HS and NLS cost functions are illustrated. When more than approximately two periods or more are assumed to be in a segment, the two cost functions are nearly the same whereas they become more and more distinct for a decreasing fundamental frequency. When the fundamental frequency is low and the data are real-valued, an error is also made if estimators based on the complex-valued harmonic model are used instead of estimators based on the real-valued harmonic model. The error is introduced when the real-valued signal is converted into an analytic signal (complex-valued) by use of the Hilbert transform which ignores the interaction effects that occur between positive and negative frequency components when the fundamental frequency is low. As demonstrated in [21], a much better estimation accuracy is obtained for low fundamental frequencies if the NLS estimators for the real-valued signal model is used instead of the NLS estimator for the complex-valued one.

Although the NLS estimator has been known for at least 25 years and has some very

attractive properties, which have been investigated thoroughly in, e.g., [2, 17, 21–24], we are not aware of any fast implementations of it. In fact, we believe that one of the main reasons for the popularity of the HS method is that it shares many desirable properties with the NLS method, but is much more computationally efficient. In this paper, however, we show that the evaluation of the NLS cost function can be reduced to the same order of time complexity as that of the HS method. More precisely, we show that we can reduce the cost of evaluating the NLS cost function on an F-point grid for all candidate model orders $l = 1, \ldots, L$ from $\mathcal{O}(F \log F) + \mathcal{O}(FL^3)$ to $\mathcal{O}(F \log F) + \mathcal{O}(FL)$ which is the same as that for the HS method. In addition to making each cost function evaluation as cheap as possible, we also derive how the number of grid points F depends on the segment length N and the maximum candidate model order L. This result is important to ensure that we neither over- nor undersample the cost function and can also be used to make the HS method faster.

The rest of this paper is organised as follows. In Sec. 2, we first introduce the signal model, the ML estimator, the NLS cost function, and the HS method. We also show how they are related to each other. Then, the standard way of computing the NLS and HS cost functions are described in Sec. 3. To speed up the computation of the NLS cost function, we describe how the number of cost function evaluations are minimised in Sec. 4 and how each cost function evaluation can be made efficiently in Sec. 5. Finally, we investigate how fine the cost function grid should be and quantify the computational savings using Monte Carlo simulations in Sec. 6.

2. Signal Model and NLS Cost Function

The real-valued signal model for a uniformly sampled and periodic signal in additive noise $e(n)$ is given by

$$x(n) = \sum_{i=1}^{l} \left[a_i \cos(i\omega_0 n) - b_i \sin(i\omega_0 n)\right] + e(n) \tag{1}$$

where a_i and b_i are the linear weights of the i'th harmonic component and ω_0 is the fundamental frequency in radians per sample. If an N-dimensional data set $\{x(n)\}_{n=n_0}^{n_0+N-1}$ is observed, the signal model can be written in vector form as

$$\boldsymbol{x} = \boldsymbol{Z}_l(\omega_0)\boldsymbol{\alpha}_l + \boldsymbol{e} \tag{2}$$

where we have defined

$$\boldsymbol{x} = \begin{bmatrix} x(n_0) & \cdots & x(n_0+N-1) \end{bmatrix}^T \tag{3}$$

$$\boldsymbol{e} = \begin{bmatrix} e(n_0) & \cdots & e(n_0+N-1) \end{bmatrix}^T \tag{4}$$

$$\boldsymbol{Z}_l(\omega) = \begin{bmatrix} \boldsymbol{C}_l(\omega) & \boldsymbol{S}_l(\omega) \end{bmatrix} \tag{5}$$

$$\boldsymbol{C}_l(\omega) = \begin{bmatrix} \boldsymbol{c}(\omega) & \boldsymbol{c}(2\omega) & \cdots & \boldsymbol{c}(l\omega) \end{bmatrix} \tag{6}$$

$$\boldsymbol{S}_l(\omega) = \begin{bmatrix} \boldsymbol{s}(\omega) & \boldsymbol{s}(2\omega) & \cdots & \boldsymbol{s}(l\omega) \end{bmatrix} \tag{7}$$

$$\boldsymbol{c}(\omega) = \begin{bmatrix} \cos(\omega n_0) & \cdots & \cos(\omega(n_0+N-1)) \end{bmatrix}^T \tag{8}$$

$$\boldsymbol{s}(\omega) = \begin{bmatrix} \sin(\omega n_0) & \cdots & \sin(\omega(n_0+N-1)) \end{bmatrix}^T \tag{9}$$

$$\boldsymbol{\alpha}_l = \begin{bmatrix} \boldsymbol{a}_l^T & -\boldsymbol{b}_l^T \end{bmatrix}^T \tag{10}$$

$$\boldsymbol{a}_l = \begin{bmatrix} a_1 & \cdots & a_l \end{bmatrix}^T \tag{11}$$

$$\boldsymbol{b}_l = \begin{bmatrix} b_1 & \cdots & b_l \end{bmatrix}^T . \tag{12}$$

Unless we are interested in estimating the phases of the sinusoidal components, the start index n_0 can be selected arbitrarily. Often, however, it is set to $n_0 = 0$ for notational convenience, but, as we demonstrate in this paper, it is advantageous to set it to $n_0 = -(N-1)/2$ to lower the computational complexity of computing the NLS cost function.

The maximum likelihood (ML) estimator of the fundamental frequency is statistically efficient asymptotically. That is, it has the optimal estimation accuracy when enough data are available. When the noise $\boldsymbol{e}$ in (2) is assumed to be white and Gaussian, the ML estimator of $\boldsymbol{\alpha}_l$ and ω_0 are given by

$$(\hat{\boldsymbol{\alpha}}_l, \hat{\omega}_0) = \underset{\omega_0 \in \Omega_l, \boldsymbol{\alpha}_l \in \mathbb{R}^{2l}}{\operatorname{argmin}} \|\boldsymbol{x} - \boldsymbol{Z}_l(\omega_0)\boldsymbol{\alpha}_l\|_2^2 \tag{13}$$

where Ω_l and $\mathbb{R}$ are a finite interval on $(0, \pi)$ and the set of real numbers, respectively. By substituting the ML estimate

$$\hat{\boldsymbol{\alpha}}_l(\omega_0) = \left[\boldsymbol{Z}_l^T(\omega_0)\boldsymbol{Z}_l(\omega_0)\right]^{-1} \boldsymbol{Z}_l^T(\omega_0)\boldsymbol{x} \tag{14}$$

back into the above optimisation problem, we obtain the ML estimator for the fundamental frequency as

$$\hat{\omega}_0 = \underset{\omega_0 \in \Omega_l}{\operatorname{argmax}} J_{\mathrm{NLS}}(\omega_0, l) \tag{15}$$

where

$$J_{\mathrm{NLS}}(\omega_0, l) = \boldsymbol{x}^T \boldsymbol{Z}_l(\omega_0) \left[\boldsymbol{Z}_l^T(\omega_0) \boldsymbol{Z}_l(\omega_0) \right]^{-1} \boldsymbol{Z}_l^T(\omega_0) \boldsymbol{x} \tag{16}$$

is the non-linear least squares (NLS) cost function. We term the cost function the NLS cost function since the ML estimator above produces the same estimates as the NLS estimator, and the main contribution of this paper is an algorithm that can produce these NLS estimates in a computationally efficient manner. In the HS method [19, 20], the matrix inversion in the NLS cost function is avoided by replacing $\boldsymbol{Z}_l^T(\omega_0)\boldsymbol{Z}_l(\omega_0)$ by a scaled identity matrix. This approximation is based on that sinusoids are asymptotically orthogonal in N, i.e., that

$$\lim_{N\to\infty} 2N^{-1} \boldsymbol{Z}_l^T(\omega_0)\boldsymbol{Z}_l(\omega_0) = \boldsymbol{I}_{2l} \tag{17}$$

where $\boldsymbol{I}_{2l}$ is the $2l \times 2l$ identity matrix. Thus, the HS cost function is given by

$$J_{\mathrm{HS}}(\omega_0, l) = \| \boldsymbol{Z}_l^T(\omega_0)\boldsymbol{x} \|_2^2 \ . \tag{18}$$

An example of the two cost functions are given in Fig. 1, and we see that they are very similar, unless the fundamental frequency is low. Note that the equality in (17) holds for a finite N if ω_0 corresponds to an integer number of cycles in the segment of data, i.e., that $\omega_0 = 2\pi k/N$ where k is an integer or, equivalently, that $\boldsymbol{Z}_l(\omega_0)$ has orthogonal columns. This can also be observed in Fig. 1 where the two cost functions intersect at integer numbers of cycles per segment.

3. Standard Algorithm

As alluded to in the introduction, we focus on computing the NLS cost function in (16) efficiently over a uniform grid[3] $\mathbb{Q}_l = \{2\pi(f-1)/F\}_{f=2}^{\lceil F/(2l)\rceil}$ consisting of $\lceil F/(2l)\rceil - 1$ points for all model orders $l \in \{1, \ldots, L\}$. That is, the first step (for all candidate model orders) of the three step procedure described in the introduction. The typical way of doing this is outlined in Algorithm 1. Depending on the size of F and L, the main contributions to the computational complexity are either the FFT computation in line 1

[3]The grid point corresponding to $f = 1$ is not included since the NLS cost function is not defined for $\omega_0 = 0$.

Algorithm 1 The standard algorithm for computing the NLS cost function matrix $[\boldsymbol{J}]_{l,f} = J_{\text{NLS}}(\omega_0, l)$ where the row indices are the L model orders and the column indices are the F fundamental frequencies on a uniform grid on $[0, 2\pi)$. The scalars f and ω_0 are related as in line 5. The notation $\odot$ and $[\cdot]_{i,k}$ denotes element-wise multiplication and element (i, k), respectively. Note that the vector function $\boldsymbol{m}_l(\omega_0, \boldsymbol{f})$ is a selection function that forms the vector $\boldsymbol{w}_l(\omega_0)$ as described in Sec. 5.1.

1: $\boldsymbol{f} = \text{fft}(\boldsymbol{x})$ ▷ $\mathcal{O}(F \log F)$
2: $[\boldsymbol{J}]_{1,1:F} = 2N^{-1}(\boldsymbol{f}^* \odot \boldsymbol{f})^T$ ▷ $\mathcal{O}(F)$
3: **for** $l \in \{2, 3, \ldots, L\}$ **do**
4: **for** $f \in \{2, 3, \ldots, \lceil F/(2l) \rceil\}$ **do**
5: $\omega_0 = 2\pi(f-1)/F$ ▷ $\mathcal{O}(1)$
6: $\boldsymbol{w}_l(\omega_0) = \boldsymbol{m}_l(\omega_0, \boldsymbol{f})$ ▷ $\mathcal{O}(1)$
7: Solve $\boldsymbol{Z}_l^T(\omega_0)\boldsymbol{Z}_l(\omega_0)\boldsymbol{\alpha}_l = \boldsymbol{w}_l(\omega_0)$ for $\boldsymbol{\alpha}_l$ ▷ $\mathcal{O}(l^3)$
8: $[\boldsymbol{J}]_{l,f} = \boldsymbol{w}_l^T(\omega_0)\boldsymbol{\alpha}_l$ ▷ $\mathcal{O}(l)$
9: **end for**
10: **end for**

or the computation of the solution to the linear system in line 7 which for all frequencies in $\mathbb{Q}_l$ and all model orders in $\{1, \ldots, L\}$ has time complexity in the order of

$$\sum_{l=1}^{L} \sum_{f=2}^{\lceil F/(2l) \rceil} \mathcal{O}(l^3) = \mathcal{O}(FL^3) \ . \tag{19}$$

Thus, the time complexity of the standard algorithm is $\mathcal{O}(F \log F) + \mathcal{O}(FL^3)$. Typically, however, the cost of solving the system in line 7 dominates the total cost, and this is precisely the reason why the HS method is so popular. Due to the approximation based on the limit in (17), line 7 in Algorithm 1 can be replaced by the much simpler problem

$$(N/2)\boldsymbol{\alpha}_l = \boldsymbol{w}_l \tag{20}$$

which can be solved in linear time complexity. In all other aspects, the HS method is identical to the standard algorithm. The time complexity of the harmonic summation method is $\mathcal{O}(F \log F) + \mathcal{O}(FL)$. Later in this paper, we show how the linear problem in line 7 of algorithm 1 can be solved exactly in linear time complexity by exploiting matrix

structures and by using recursive updates. First, however, we establish how the number of grid points F should be selected.

4. Selecting the Grid Size

Selecting the grid size is important so that we neither make the grid so fine that we make too many unnecessary cost function evaluations nor so coarse that we undersample and miss the global maximum. For a given data set, one can always experiment with the grid size to find a good value for F, but we would like to be able to say something about the grid size before seeing the data. In this section, we do exactly that.

Suppose we wish to set the grid size $\Delta\omega_0$ so that the cost function has decreased by a factor of $g > 1$ when we move $\pm\Delta\omega_0$ away from the maximiser $\hat{\omega}_0$. That is,

$$J_{\mathrm{NLS}}(\hat{\omega}_0 \pm \Delta\omega_0, l) = \frac{J_{\mathrm{NLS}}(\hat{\omega}_0, l)}{g} \ . \tag{21}$$

Unfortunately, this equation cannot be solved for $\Delta\omega_0$. However, by replacing $J_{\mathrm{NLS}}(\hat{\omega}_0 \pm \Delta\omega_0, l)$ with its second order Taylor approximation around the maximiser $\hat{\omega}_0$, we obtain

$$J_{\mathrm{NLS}}(\hat{\omega}_0, l) + \frac{\Delta\omega_0^2}{2} H(\hat{\omega}_0, l) \approx \frac{J_{\mathrm{NLS}}(\hat{\omega}_0, l)}{g} \tag{22}$$

where $H(\omega_0, l)$ is the second order derivative of $J_{\mathrm{NLS}}(\omega_0, l)$ w.r.t. ω_0. This equation can easily be solved for $\Delta\omega_0$ giving the approximate solution

$$\Delta\omega_0 \approx \sqrt{2\frac{1-g}{g}\frac{J_{\mathrm{NLS}}(\hat{\omega}_0, l)}{H(\hat{\omega}_0, l)}} \ . \tag{23}$$

The expression for $H(\hat{\omega}_0, l)$ is very complicated [25], but if we make the same approximation as in the HS method (see (17)) and assume a noise free signal, we can find a much simpler expression for the grid size. Specifically, we obtain that [24]

$$J_{\mathrm{NLS}}(\hat{\omega}_0, l) \approx \frac{N}{2} \sum_{i=1}^{l} \hat{A}_i^2(\hat{\omega}_0) \tag{24}$$

$$H(\hat{\omega}_0, l) \approx -\frac{N^3}{12} \sum_{i=1}^{l} \hat{A}_i^2(\hat{\omega}_0) i^2 \tag{25}$$

where the squared amplitudes are given by $\hat{A}_i^2(\hat{\omega}_0) = \hat{a}_i^2(\hat{\omega}_0) + \hat{b}_i^2(\hat{\omega}_0)$. Inserting these into (23) give

$$\Delta\omega_0 \approx \sqrt{12\frac{g-1}{g}\frac{\sum_{i=1}^{l}\hat{A}_i^2(\hat{\omega}_0)}{N^2\sum_{i=1}^{l}\hat{A}_i^2(\hat{\omega}_0)i^2}} \ . \tag{26}$$

To avoid that the grid size depends on the estimated amplitudes, we compute a lower bound on the grid size by selecting $\{\hat{A}_i(\hat{\omega}_0)\}_{i=1}^{l}$ so that

$$\begin{aligned} \{\tilde{A}_1(\hat{\omega}_0), \cdots, \tilde{A}_l(\hat{\omega}_0)\} = \ & \underset{\text{subject to}}{\text{argmax}} \quad \sum_{i=1}^{l}\hat{A}_i^2(\hat{\omega}_0)i^2 \\ & \qquad\quad \sum_{i=1}^{l}\hat{A}_i^2(\hat{\omega}_0) = \gamma^2 \\ & \qquad\quad \hat{A}_i(\hat{\omega}_0) \geq 0 \end{aligned} \ . \tag{27}$$

By differentiating the Lagrangian of the problem w.r.t. $\hat{\boldsymbol{\alpha}}(\hat{\omega}_0)$, we see that $\tilde{\boldsymbol{\alpha}}(\hat{\omega}_0)$ is related to the eigenvector pertaining to the maximum eigenvalue of the matrix $\text{diag}(1, \ldots, l)^2$ by any real-valued factor γ. Thus, we get that

$$\tilde{A}_i(\hat{\omega}_0) = \begin{cases} |\gamma| & \text{for } i = l \\ 0 & \text{otherwise} \end{cases} \ . \tag{28}$$

Inserting these values for $\{\hat{A}_i(\hat{\omega}_0)\}_{i=1}^{l}$ in (26) now readily gives

$$\Delta\omega_0 > \sqrt{12\frac{g-1}{g}}\frac{1}{Nl} \ . \tag{29}$$

Thus, the grid size depends on both the number of data points N and the model order l. Please also note the following:

- Since the model order l is usually unknown, we just set it to the maximum candidate model order L when finding a value for $\Delta\omega_0$.
- From a computational perspective, it is much more efficient to evaluate the cost function on a coarse grid followed by a refinement step than to evaluate the cost function on only a fine grid [26, 27]. The coarseness of the grid is controlled with the scalar g, and we will find a suitable value for it in the Sec 6.
- In the above derivation, we made the approximation based on (17). Despite this, we have found through simulations that the result in (29) is also useful for a low fundamental frequency. This is also demonstrated in the simulation section.

5. Fast Algorithm

In the previous section, we explained how the minimum number of cost function evaluations can be predicted by selecting an appropriate grid size. In this section, we reduce the cost associated with each cost function evaluation of the NLS method so that the total time complexity is reduced to the same order as that of the HS method. We have previously outlined these ideas in [14], but we here give a much more detailed description of how the time complexity is reduced. This reduction is based on five key facts which are described in the following five sections.

5.1. Solving the linear system efficiently

To make an efficient implementation of the NLS cost function, we have to solve line 7 of Algorithm 1 efficiently. That is, we have to solve

$$\boldsymbol{A}_l(\omega_0)\boldsymbol{\alpha}_l = \boldsymbol{w}_l(\omega_0) \tag{30}$$

for $\boldsymbol{\alpha}_l$ in an efficient way where

$$\boldsymbol{A}_l(\omega_0) = \boldsymbol{Z}_l^T(\omega_0)\boldsymbol{Z}_l(\omega_0) \tag{31}$$

$$\boldsymbol{w}_l(\omega_0) = \boldsymbol{Z}_l^T(\omega_0)\boldsymbol{x}\ . \tag{32}$$

By using the definitions in (5)–(9), we see that the right-hand side of the linear system involves terms of the form

$$\boldsymbol{c}^T(\omega_0 l)\boldsymbol{x} = \Re\Big[\exp\left(-j\omega_0 l n_0\right)\sum_{n=0}^{N-1} x(n+n_0)\exp\left(-j\omega_0 l n\right)\Big] \tag{33}$$

$$\boldsymbol{s}^T(\omega_0 l)\boldsymbol{x} = -\Im\Big[\exp\left(-j\omega_0 l n_0\right)\sum_{n=0}^{N-1} x(n+n_0)\exp\left(-j\omega_0 l n\right)\Big]\,. \tag{34}$$

These can be computed efficiently using a standard FFT algorithm since $\omega_0 \in \mathbb{Q}_l$.

5.2. The structure of $\boldsymbol{A}_l(\omega_0)$

From the definitions in (5)–(9), we see that the matrix $\boldsymbol{A}_l(\omega_0)$ consists of four terms given by

$$\boldsymbol{c}^T(i\omega_0)\boldsymbol{c}(k\omega_0) = t_{i-k}(\omega_0) + h_{i+k}(\omega_0) \tag{35}$$

$$\boldsymbol{s}^T(i\omega_0)\boldsymbol{s}(k\omega_0) = t_{i-k}(\omega_0) - h_{i+k}(\omega_0) \tag{36}$$

$$\boldsymbol{s}^T(i\omega_0)\boldsymbol{c}(k\omega_0) = \tilde{t}_{i-k}(\omega_0) + \tilde{h}_{i+k}(\omega_0) \tag{37}$$

$$\boldsymbol{c}^T(i\omega_0)\boldsymbol{s}(k\omega_0) = -\tilde{t}_{i-k}(\omega_0) + \tilde{h}_{i+k}(\omega_0)\ . \tag{38}$$

The terms on the right hand sides of these equations are defined for $i, k \in \{1, 2, \ldots, l\}$ as

$$t_{i-k}(\omega_0) = t_{k-i}(\omega_0) = \begin{cases} \frac{1}{2}\cos\left(\omega_0(i-k)\left[n_0 + \frac{N-1}{2}\right]\right) & \\ \quad \times \frac{\sin(\omega_0(i-k)N/2)}{\sin(\omega_0(i-k)/2)} & i \neq k \\ N/2 & i = k \end{cases} \tag{39}$$

$$h_{i+k}(\omega_0) = \frac{1}{2}\cos\left(\omega_0(i+k)\left[n_0 + \frac{N-1}{2}\right]\right)\frac{\sin(\omega_0(i+k)N/2)}{\sin(\omega_0(i+k)/2)} \tag{40}$$

$$\tilde{t}_{i-k}(\omega_0) = -\tilde{t}_{k-i}(\omega_0) = \begin{cases} \frac{1}{2}\sin\left(\omega_0(i-k)\left[n_0 + \frac{N-1}{2}\right]\right) & \\ \quad \times \frac{\sin(\omega_0(i-k)N/2)}{\sin(\omega_0(i-k)/2)} & i \neq k \\ 0 & i = k \end{cases} \tag{41}$$

$$\tilde{h}_{i+k}(\omega_0) = \frac{1}{2}\sin\left(\omega_0(i+k)\left[n_0 + \frac{N-1}{2}\right]\right)\frac{\sin(\omega_0(i+k)N/2)}{\sin(\omega_0(i+k)/2)}\ . \tag{42}$$

Using these definitions, the matrix $\boldsymbol{A}_l(\omega_0)$ can be partitioned as

$$\boldsymbol{A}_l(\omega_0) = \begin{bmatrix} \boldsymbol{T}_l(\omega_0) & -\tilde{\boldsymbol{T}}_l(\omega_0) \\ \tilde{\boldsymbol{T}}_l(\omega_0) & \boldsymbol{T}_l(\omega_0) \end{bmatrix} + \begin{bmatrix} \boldsymbol{H}_l(\omega_0) & \tilde{\boldsymbol{H}}_l(\omega_0) \\ \tilde{\boldsymbol{H}}_l(\omega_0) & -\boldsymbol{H}_l(\omega_0) \end{bmatrix} \tag{43}$$

where

$$\boldsymbol{T}_l(\omega_0) = \begin{bmatrix} t_0(\omega_0) & t_1(\omega_0) & \cdots & t_{l-1}(\omega_0) \\ t_1(\omega_0) & t_0(\omega_0) & \cdots & t_{l-2}(\omega_0) \\ \vdots & \vdots & \ddots & \vdots \\ t_{l-1}(\omega_0) & t_{l-2}(\omega_0) & \cdots & t_0(\omega_0) \end{bmatrix} \tag{44}$$

$$\boldsymbol{H}_l(\omega_0) = \begin{bmatrix} h_2(\omega_0) & h_3(\omega_0) & \cdots & h_{l+1}(\omega_0) \\ h_3(\omega_0) & h_4(\omega_0) & \iddots & h_{l+2}(\omega_0) \\ \vdots & \iddots & \iddots & \vdots \\ h_{l+1}(\omega_0) & h_{l+2}(\omega_0) & \cdots & h_{2l}(\omega_0) \end{bmatrix} \tag{45}$$

and $\tilde{\boldsymbol{T}}_l(\omega_0)$ and $\tilde{\boldsymbol{H}}_l(\omega_0)$ are defined similarly to $\boldsymbol{T}_l(\omega_0)$ and $\boldsymbol{H}_l(\omega_0)$, respectively. Note that $\boldsymbol{T}_l(\omega_0)$ and $\tilde{\boldsymbol{T}}_l(\omega_0)$ are Toeplitz matrices while $\boldsymbol{H}_l(\omega_0)$ and $\tilde{\boldsymbol{H}}_l(\omega_0)$ are Hankel matrices.

5.3. Selection of the start index n_0

The value of $J_{\mathrm{NLS}}(\omega_0, l)$ does *not* depend on the value of n_0. A convenient choice is then $n_0 = -\frac{N-1}{2}$ since this makes $\tilde{t}_{i-k}(\omega_0) = \tilde{h}_{i+k}(\omega_0) = 0$ for $i, k = 1, \ldots, l$ and any $\omega_0 \in \Omega_l$. The linear system in (30) is then separable so that it can be written as the two systems

$$(\boldsymbol{T}_l(\omega_0) + \boldsymbol{H}_l(\omega_0))\boldsymbol{a}_l(\omega_0) = \bar{\boldsymbol{w}}_l(\omega_0) \tag{46}$$

$$(\boldsymbol{T}_l(\omega_0) - \boldsymbol{H}_l(\omega_0))\boldsymbol{b}_l(\omega_0) = -\tilde{\boldsymbol{w}}_l(\omega_0) \tag{47}$$

where $\boldsymbol{w}_l(\omega_0) = \begin{bmatrix} \bar{\boldsymbol{w}}_l^T(\omega_0) & \tilde{\boldsymbol{w}}_l^T(\omega_0) \end{bmatrix}^T$ is partitioned as in (43).

5.4. Solving the two linear systems for a given model order

The two linear systems in (46) and (47) both have a Toeplitz-plus-Hankel structure which can be solved efficiently with time complexity $\mathcal{O}(l^2)$ using algorithms such as those suggested in [28, 29]. Problems involving a Toeplitz-plus-Hankel structure are commonly seen as a special case of a larger class of problems having displacement structure (see, e.g., the overview work in [30]).

5.5. Solving the two linear systems for all model orders

Not only are the systems Toeplitz-plus-Hankel, but any upper-left subsystem is a solution to the linear system with a lower order. Specifically, if we define

$$\boldsymbol{R}_l(\omega_0) = \boldsymbol{T}_l(\omega_0) + \boldsymbol{H}_l(\omega_0) \tag{48}$$

$$\boldsymbol{r}_l(\omega_0) = \begin{bmatrix} t_l(\omega_0) + h_{l+2}(\omega_0) \\ t_{l-1}(\omega_0) + h_{l+3}(\omega_0) \\ \vdots \\ t_1(\omega_0) + h_{2l+1}(\omega_0) \end{bmatrix} \tag{49}$$

$$r_l(\omega_0) = t_0(\omega_0) + h_{2l+2}(\omega_0)\ , \tag{50}$$

the system matrix and the data vector in (46) of order $l+1$ are given by

$$\boldsymbol{R}_{l+1}(\omega_0) = \begin{bmatrix} \boldsymbol{R}_l(\omega_0) & \boldsymbol{r}_l(\omega_0) \\ \boldsymbol{r}_l^T(\omega_0) & r_l(\omega_0) \end{bmatrix} \tag{51}$$

$$\bar{\boldsymbol{w}}_{l+1}(\omega_0) = \begin{bmatrix} \bar{\boldsymbol{w}}_l(\omega_0) \\ \boldsymbol{c}^T((l+1)\omega_0)\boldsymbol{x} \end{bmatrix}\ . \tag{52}$$

This implies that it is possible to calculate the solution to

$$\boldsymbol{R}_L(\omega_0)\boldsymbol{a}_L(\omega_0) = \bar{\boldsymbol{w}}_L(\omega_0) \tag{53}$$

recursive-in-order using the systems formed by $\boldsymbol{R}_l(\omega_0), \bar{\boldsymbol{w}}_l(\omega_0)$ for $l = 1, \ldots, L-1$. Moreover, we obtain the solutions $\boldsymbol{a}_l(\omega_0), l = 1, \ldots, L-1$ as well in the process. The same idea can be exploited for solving the linear system in (47).

5.6. Solving a Toeplitz-plus-Hankel system

By using these five key facts, we find that the time complexity for solving the linear system in line 7 of Algorithm 1 for all frequencies and model orders is

$$\sum_{l=1}^{L} \sum_{f=2}^{\lceil F/(2l) \rceil} \mathcal{O}(l) = \mathcal{O}(FL)\,. \tag{54}$$

Thus, the total time complexity of the proposed algorithm is $\mathcal{O}(F \log F) + \mathcal{O}(FL)$ for computing the exact NLS cost function over a uniform grid for all candidate model orders

up to L. The reduction is obtained by exploiting the two Toeplitz-plus-Hankel systems in (46) and (47). Since these systems have the same structure, they can be solved using the same method. In the rest of this section, we therefore only focus on presenting an efficient algorithm for solving (46) recursive-in-order. The presented algorithm is the recursive algorithm in [28] of which we here outline the main ideas behind for completeness and since the algorithm is rather involved. Note that there exist Toeplitz-plus-Hankel solvers which do not solve (46) recursive-in-order due to a reformulation of the original problem [31]. To simplify the notation, we omit the ω_0 dependency in the following.

5.6.1. The data dependent step

The presented algorithm for solving a Toeplitz-plus-Hankel system can be divided in a data dependent and a data independent step. In the data dependent step, we assume that we have computed the solution to

$$\boldsymbol{R}_l \boldsymbol{\gamma}_l = \boldsymbol{e}_l, \quad l = 1, \ldots, L \tag{55}$$

for the data independent step where $\boldsymbol{e}_l = \begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix}^T \in \mathbb{R}^{l \times 1}$ and $\boldsymbol{\gamma}_l \in \mathbb{R}^{l \times 1}$. The algorithm then computes the solution to $\boldsymbol{R}_l \boldsymbol{a}_l = \bar{\boldsymbol{w}}_l$ using the recursive updates

$$\lambda_l = [\bar{\boldsymbol{w}}_L]_l - \boldsymbol{r}_l^T \boldsymbol{a}_{l-1} \tag{56}$$

$$\boldsymbol{a}_l = \begin{bmatrix} \boldsymbol{a}_{l-1} \\ 0 \end{bmatrix} + \lambda_l \boldsymbol{\gamma}_l \ . \tag{57}$$

5.6.2. The data independent step

In the data independent step, we focus on solving

$$\boldsymbol{R}_{l+1} \boldsymbol{\gamma}_{l+1} = \boldsymbol{e}_{l+1} \tag{58}$$

efficiently given the solution $\boldsymbol{\gamma}_l$ to $\boldsymbol{R}_l \boldsymbol{\gamma}_l = \boldsymbol{e}_l$. Due to the Toeplitz-plus-Hankel structure of $\boldsymbol{R}_l$, we have that [28]

$$(\boldsymbol{L}_l + \boldsymbol{L}_l^T)\boldsymbol{R}_l - \boldsymbol{R}_l(\boldsymbol{L}_l + \boldsymbol{L}_l^T) = \boldsymbol{q}_l \boldsymbol{e}_1^T - \boldsymbol{e}_1 \boldsymbol{q}_l^T + \boldsymbol{r}_l \boldsymbol{e}_l^T - \boldsymbol{e}_l \boldsymbol{r}_l^T \tag{59}$$

where $\boldsymbol{L}_l$ is a lower triangular shift matrix of size $l \times l$

$$\boldsymbol{L}_l = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 1 & 0 \end{bmatrix} . \tag{60}$$

Moreover, $\boldsymbol{R}_l$ and $\boldsymbol{r}_l$ can be found from (48) and (49), respectively, and

$$\boldsymbol{e}_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix}^T \in \mathbb{R}^{l \times 1} \tag{61}$$

$$\boldsymbol{q}_l = \begin{bmatrix} t_1 & t_2 + h_2 & \cdots & t_l + h_l \end{bmatrix}^T \in \mathbb{R}^{l \times 1} \tag{62}$$

for $l = 1, \ldots, L-1$. Multiplying (59) with $\boldsymbol{R}_l^{-1}$ from both the left and the right gives

$$\boldsymbol{R}_l^{-1}(\boldsymbol{L}_l + \boldsymbol{L}_l^T) - (\boldsymbol{L}_l + \boldsymbol{L}_l^T)\boldsymbol{R}_l^{-1} = \boldsymbol{\phi}_l\boldsymbol{\psi}_l^T - \boldsymbol{\psi}_l\boldsymbol{\phi}_l^T - \boldsymbol{\rho}_l\boldsymbol{\gamma}_l^T + \boldsymbol{\gamma}_l\boldsymbol{\rho}_l^T \tag{63}$$

where $\boldsymbol{\phi}_l$, $\boldsymbol{\psi}_l$, and $\boldsymbol{\rho}_l$ are the solutions to

$$\boldsymbol{R}_l\boldsymbol{\phi}_l = \boldsymbol{q}_l, \tag{64}$$

$$\boldsymbol{R}_l\boldsymbol{\psi}_l = \boldsymbol{e}_1, \tag{65}$$

$$\boldsymbol{R}_l\boldsymbol{\rho}_l = -\boldsymbol{r}_l, \tag{66}$$

235 respectively for $l = 1, \ldots, L-1$. If we now multiply (63) from the right with $\boldsymbol{e}_l$ and solve for the second last term on the right hand side, we obtain that

$$[\boldsymbol{\gamma}_l]_l\boldsymbol{\rho}_l = (\boldsymbol{L}_l + \boldsymbol{L}_l^T)\boldsymbol{\gamma}_l - \boldsymbol{R}_l^{-1}\begin{bmatrix} \boldsymbol{e}_{l-1} \\ 0 \end{bmatrix} + [\boldsymbol{\rho}_l]_l\boldsymbol{\gamma}_l + [\boldsymbol{\psi}_l]_l\boldsymbol{\phi}_l - [\boldsymbol{\phi}_l]_l\boldsymbol{\psi}_l \triangleq \boldsymbol{\beta}_l . \tag{67}$$

The vector $\boldsymbol{\rho}_l$ can be expressed in terms of the elements of $\boldsymbol{\gamma}_{l+1}$. To derive this relationship, we write (58) using (51) as

$$\begin{bmatrix} \boldsymbol{R}_l & \boldsymbol{r}_l \\ \boldsymbol{r}_l^T & r_l \end{bmatrix}\begin{bmatrix} [\boldsymbol{\gamma}_{l+1}]_{1:l} \\ [\boldsymbol{\gamma}_{l+1}]_{l+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ 1 \end{bmatrix} = \boldsymbol{e}_{l+1} . \tag{68}$$

These equations can also be written as

$$\boldsymbol{R}_l^{-1}\boldsymbol{r}_l = -\boldsymbol{\rho}_l = -\frac{[\boldsymbol{\gamma}_{l+1}]_{1:l}}{[\boldsymbol{\gamma}_{l+1}]_{l+1}} \tag{69}$$

$$[\boldsymbol{\gamma}_{l+1}]_{l+1} = \left(r_l + \boldsymbol{r}_l^T\boldsymbol{\rho}_l\right)^{-1} . \tag{70}$$

By combining (67), (69), and (70), we now get linear complexity updates of $[\boldsymbol{\gamma}_{l+1}]_{1:l}$ and $[\boldsymbol{\gamma}_{l+1}]_{l+1}$ given by

$$[\boldsymbol{\gamma}_{l+1}]_{l+1} = \left(r_l + \frac{\boldsymbol{r}_l^T \boldsymbol{\beta}_l}{[\boldsymbol{\gamma}_l]_l} \right)^{-1} \tag{71}$$

$$[\boldsymbol{\gamma}_{l+1}]_{1:l} = \frac{[\boldsymbol{\gamma}_{l+1}]_{l+1}}{[\boldsymbol{\gamma}_l]_l} \boldsymbol{\beta}_l \ . \tag{72}$$

Both of these equations depend on $\boldsymbol{\beta}_l$ which according to (67) consists of five terms, and these can all be computed efficiently. Below, we outline how the second, fourth, and fifth term can be computed with a linear time complexity.

2. Since

$$\boldsymbol{R}_l \begin{bmatrix} \boldsymbol{\gamma}_{l-1} \\ 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{l-1} & \boldsymbol{r}_{l-1} \\ \boldsymbol{r}_{l-1}^T & r_{l-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_{l-1} \\ 0 \end{bmatrix} \tag{73}$$

$$= \begin{bmatrix} \boldsymbol{e}_{l-1} \\ 0 \end{bmatrix} + \boldsymbol{r}_{l-1}^T \boldsymbol{\gamma}_{l-1} \boldsymbol{e}_l \ , \tag{74}$$

it follows that

$$\boldsymbol{R}_l^{-1} \begin{bmatrix} \boldsymbol{e}_{l-1} \\ 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\gamma}_{l-1} \\ 0 \end{bmatrix} - \boldsymbol{r}_{l-1}^T \boldsymbol{\gamma}_{l-1} \boldsymbol{\gamma}_l \ . \tag{75}$$

Note that the scalar factor in the last term is

$$\boldsymbol{r}_{l-1}^T \boldsymbol{\gamma}_{l-1} = \boldsymbol{r}_{l-1}^T \boldsymbol{R}_{l-1}^{-1} \boldsymbol{e}_{l-1} = -\boldsymbol{\rho}_{l-1}^T \boldsymbol{e}_{l-1} = -[\boldsymbol{\rho}_{l-1}]_{l-1} \ . \tag{76}$$

This proves to be useful later.

4.-5. The solution for Equation (64) and (65) can be updated in a similar way to $\boldsymbol{a}_l$ in (56)–(57)

$$\boldsymbol{\phi}_l = \begin{bmatrix} \boldsymbol{\phi}_{l-1} \\ 0 \end{bmatrix} + ([\boldsymbol{q}_l]_l - \boldsymbol{r}_{l-1}^T \boldsymbol{\phi}_{l-1}) \boldsymbol{\gamma}_l \tag{77}$$

$$\boldsymbol{\psi}_l = \begin{bmatrix} \boldsymbol{\psi}_{l-1} \\ 0 \end{bmatrix} - \boldsymbol{r}_{l-1}^T \boldsymbol{\psi}_{l-1} \boldsymbol{\gamma}_l \ . \tag{78}$$

The vector $\boldsymbol{\beta}_l$ can now be written as

$$\boldsymbol{\beta}_l = \left(([\boldsymbol{\rho}_l]_l - [\boldsymbol{\rho}_{l-1}]_{l-1}) \boldsymbol{I}_l + \boldsymbol{L}_l + \boldsymbol{L}_l^T \right) \boldsymbol{\gamma}_l - \begin{bmatrix} \boldsymbol{\gamma}_{l-1} \\ 0 \end{bmatrix} + [\boldsymbol{\psi}_l]_l \boldsymbol{\phi}_l - [\boldsymbol{\phi}_l]_l \boldsymbol{\psi}_l \ . \tag{79}$$

Thus, given $\boldsymbol{\gamma}_l$, $\boldsymbol{R}_l$, $\boldsymbol{r}_l$, r_l, $\boldsymbol{q}_l$, and $[\boldsymbol{\rho}_{l-1}]_{l-1}$, we can compute the solution to (58) with linear time complexity using the following steps.

1. Compute $\boldsymbol{\phi}_l$ and $\boldsymbol{\psi}_l$ using (77) and (78).
2. Compute $[\boldsymbol{\rho}_l]_l$ using (76).
3. Compute $\boldsymbol{\beta}_l$ using (79).
4. Compute $\boldsymbol{\gamma}_{l+1}$ using (71) and (72).

This algorithm is detailed in Algorithm 2. Note the following about the algorithm.

- The outer for-loop in Algorithm 2 is only there to keep the notation simpler. In practice, the outer loop can be vectorised and this is also done in our MATLAB implementation to speed-up execution. The index k is also only introduced to keep the indexing simpler.

- In the algorithm, the range of candidate fundamental frequencies is the maximum one. In our MATLAB implementations both a lower and an upper bound can be specified for the fundamental frequency. Note that including prior knowledge on the fundamental frequency implies that the ML estimator should be referred to as a (maximum aposteriori) MAP estimator instead.

- The algorithm for solving (47) instead of (46) is nearly identical to Algorithm 2. One obvious difference is that the data vector $\tilde{\boldsymbol{w}}_l$ should be used instead of $\bar{\boldsymbol{w}}_l$. Another difference is that the vector $\boldsymbol{q}_L$ in (62) now is given by

$$\boldsymbol{q}_L = \begin{bmatrix} t_1 & 0 & \cdots & 0 \end{bmatrix}^T \in \mathbb{R}^{L \times 1} \tag{80}$$

  since $t_l - h_l = 0$. From (64) and (65), it therefore follows that $\boldsymbol{\phi}_l = t_1 \boldsymbol{\psi}_l$ so

$$0 = [\boldsymbol{\psi}_l]_l \boldsymbol{\phi}_l - [\boldsymbol{\phi}_l]_l \boldsymbol{\psi}_l \ . \tag{81}$$

  Thus, neither $\boldsymbol{\phi}_l$ nor $\boldsymbol{\psi}_l$ has to be computed.

- Since the data independent step of the algorithm does not depend on the data, the value of $\boldsymbol{\gamma}_l(\omega_0)$ for all pairs of candidate fundamental frequencies and model orders can be computed offline and stored in memory. Only the data dependent step then has to be computed which may be of interest in, e.g., real-time applications.

The lines pertaining to the data dependent step have been marked by a star in the margin of Algorithm 2. If Algorithm 3 is implemented with recursive calls of 16–27, then the space complexity of our vectorised version of the algorithm is dominated by holding the cost function $\sum_{l=1}^{\lceil F/l \rceil} F/l = \mathcal{O}(F \log L)$ and by holding intermediate vectors no longer than F in memory. Thus, the total space complexity is

$$\mathcal{O}(F \log L + F)\,. \tag{82}$$

- Algorithm 2 can easily be modified to compute the value of the NLS cost function for just a single and arbitrary frequency and model order. This is useful for speeding up the second step of the three step procedure outlined in the introduction.

- The condition number of the matrix $\boldsymbol{A}_l(\omega_0)$ follows a periodic pattern and is smallest when $\omega_0 = 2\pi k/N$ where k is an integer. In this case, the columns of $\boldsymbol{Z}_l(\omega_0)$ are orthogonal and $\boldsymbol{A}_l(\omega_0)$ is, therefore, a diagonal matrix with a condition number of 1. Between the grid points, we have emperically found that the condition number seems to be in the interval

$$1 \leq \kappa\left(\boldsymbol{A}_l(\omega_0)\right) \leq \frac{\left\lceil \frac{\omega_0 N}{2\pi} \right\rceil}{\left\lfloor \frac{\omega_0 N}{2\pi} \right\rfloor} \tag{83}$$

  where $\omega_0 N/(2\pi)$ is the number of cycles in the data segment, $\lceil \cdot \rceil$ is the ceiling operator, and $\lfloor \cdot \rfloor$ is the flooring operator. Consequently, the problem of solving the Toeplitz-plus-Hankel systems in (46) and (47) are very well conditioned unless the candidate fundamental frequency ω_0 corresponds to less than one cycle in the data segment. For such a low frequency, the condition number grows without a bound since the columns of $\boldsymbol{Z}_l(\omega_0)$ become more and more linearly dependent. Similar problems occur when the fundamental frequency $l\omega_0$ is so high that the frequency of the largest harmonic component is close to π [21]. To combat this, $\boldsymbol{A}_l(\omega_0)$ can easily be regularised by adding the scaled identity matrix $\epsilon \boldsymbol{I}_{2l}$ to it where ϵ is a small positive constant. Since such a regularisation matrix is Toeplitz and diagonal, the regularisation can easily be included in Algorithm 2 by adding ϵ to t_0. Moreover, we recommend that the regularisation parameter is made frequency dependent so that it is only non-zero for very low fundamental frequencies and for frequencies where the largest harmonic component is close to π. We note in passing that the

regularised algorithm can be viewed as an intermediate solution between the exact NLS method ($\epsilon = 0$) and the harmonic summation method ($\epsilon = \infty$). Moreover, $(\sigma^2/\epsilon)\boldsymbol{I}_{2l}$ can also in a Bayesian framework be interpreted as the variance of a zero-mean Gaussian prior on $\boldsymbol{\alpha}_l$ where σ^2 is the noise variance.

5.7. Computing the NLS cost function

Once (46) and (47) have been solved using Algorithm 2, the value of the cost function can be computed as

$$J_{\text{NLS}}(\omega_0, l) = \tilde{\boldsymbol{w}}_l^T(\omega_0)\boldsymbol{a}_l(\omega_0) - \tilde{\boldsymbol{w}}_l^T(\omega_0)\boldsymbol{b}_l(\omega_0) \ . \tag{84}$$

The algorithm for computing $J_{\text{NLS}}(\omega_0, l)$ for all candidate fundamental frequencies and model orders are summarised in Algorithm 3.

6. Simulations

In this section, we demonstrate two things. First of all, we evaluate how the grid size should be selected so that we get a grid that is neither too fine nor too coarse. We also see how well this predicted grid size works for a low fundamental frequency. Moreover, we compare the exact NLS estimator to the HS method to demonstrate how much better the former is performing than the latter for a low fundamental frequency. For comparison, we have also included the YIN estimator [11] in one of the simulations since this is one of the most popular and most cited fundamental frequency estimators. Since the contribution of this paper is a fast algorithm of a well-known estimator, we do not make a thorough evaluation of the accuracy of the estimator for different noise types or an unknown model order. Instead, we refer the interested reader to [2, 17, 21–24] and the references therein.

Secondly, and most importantly, we evaluate the proposed algorithms computational savings accessed by running benchmarks of various MATLAB implementations. The investigated algorithms are: The standard algorithm (Algorithm 1), the proposed fast algorithm (Algorithm 3) in a vectorized version, and harmonic summation. All these implementations and the code for generating the results presented here are available from `http://tinyurl.com/jknvbn`.

Algorithm 2 A fast algorithm for computing the solution to the Toeplitz-plus-Hankel linear system of equations in (46) for L model orders and F fundamental frequencies on the Fourier grid. The data dependent steps have been marked by a star in the margin.

$\boldsymbol{f} = \text{fft}(\boldsymbol{x})$ ▷ $\mathcal{O}(F \log F)^*$

for $f \in \{2, 3, \ldots, F\}$ **do**

  $\omega_0 = 2\pi(f-1)/F$ ▷ $\mathcal{O}(1)$

  $r_0 = \boldsymbol{r}_0 = t_0 + h_2$ ▷ $\mathcal{O}(1)$

  $\gamma_1 = 1/r_0$ ▷ $\mathcal{O}(1)$

  $\phi_0 = [\boldsymbol{q}_L]_1/r_0$ ▷ $\mathcal{O}(1)$

  $\psi_0 = 1/r_0$ ▷ $\mathcal{O}(1)$

  $r_1 = t_1 + h_3$ ▷ $\mathcal{O}(1)$

  $\rho_0 = -r_1/r_0$ ▷ $\mathcal{O}(1)$

  $\bar{w}_1 = \Re\{[\boldsymbol{f}]_f \exp(j\omega_0 \frac{N-1}{2})\}$ ▷ $\mathcal{O}(1)^*$

  $\lambda_1 = \bar{w}_1$ ▷ $\mathcal{O}(1)^*$

  $a_1 = \lambda_1 \gamma_1$ ▷ $\mathcal{O}(1)^*$

  for $k \in 1, 2, \ldots, L-1$ **do**

    $l = k + 1$ ▷ $\mathcal{O}(1)$

    if $f < \lceil \frac{F}{2l} \rceil$ **then** ▷ Ensure that $\omega_0 \in (0, \pi/l)$

      $\boldsymbol{\phi}_k = \begin{bmatrix} \boldsymbol{\phi}_{k-1} \\ 0 \end{bmatrix} + ([\boldsymbol{q}_k]_k - \boldsymbol{r}_{k-1}^T \boldsymbol{\phi}_{k-1})\boldsymbol{\gamma}_k$ ▷ $\mathcal{O}(l)$

      $\boldsymbol{\psi}_k = \begin{bmatrix} \boldsymbol{\psi}_{k-1} \\ 0 \end{bmatrix} - \boldsymbol{r}_{k-1}^T \boldsymbol{\psi}_{k-1} \boldsymbol{\gamma}_k$ ▷ $\mathcal{O}(l)$

      $\boldsymbol{r}_k = \begin{bmatrix} t_k + h_{k+2} & \cdots & t_1 + h_{2k+1} \end{bmatrix}^T$ ▷ $\mathcal{O}(l)$

      $[\boldsymbol{\rho}_k]_k = -\boldsymbol{r}_k^T \boldsymbol{\gamma}_k$ ▷ $\mathcal{O}(l)$

      Compute $\boldsymbol{\beta}_k$ using (79) ▷ $\mathcal{O}(l)$

      $r_{k+1} = t_0 + h_{2k+2}$ ▷ $\mathcal{O}(1)$

      $[\boldsymbol{\gamma}_{k+1}]_{k+1} = \left(r_{k+1} + \frac{\boldsymbol{r}_k^T \boldsymbol{\beta}_k}{[\boldsymbol{\gamma}_k]_k} \right)^{-1}$ ▷ $\mathcal{O}(l)$

      $[\boldsymbol{\gamma}_{k+1}]_{1:k} = \frac{[\boldsymbol{\gamma}_{k+1}]_{k+1}}{[\boldsymbol{\gamma}_k]_k} \boldsymbol{\beta}_k$ ▷ $\mathcal{O}(l)$

      $\mu_k = \exp(j\omega_0 l \frac{N-1}{2})$ ▷ $\mathcal{O}(1)$

      $[\bar{\boldsymbol{w}}_L]_{k+1} = \Re\{\mu_k [\boldsymbol{f}]_{l(f-1)+1}\}$ ▷ $\mathcal{O}(1)^*$

      $\lambda_{k+1} = [\bar{\boldsymbol{w}}_L]_{k+1} - \boldsymbol{r}_k^T \boldsymbol{a}_k$ ▷ $\mathcal{O}(l)^*$

      $\boldsymbol{a}_{k+1} = \begin{bmatrix} \boldsymbol{a}_k \\ 0 \end{bmatrix} + \lambda_{k+1} \boldsymbol{\gamma}_{k+1}$ ▷ $\mathcal{O}(l)^*$

    end if

  end for

end for

Algorithm 3 The proposed fast algorithm for computing the NLS cost function for L model orders and F fundamental frequencies on a uniform grid.

1: $\boldsymbol{f} = \text{fft}(\boldsymbol{x})$ ▷ $\mathcal{O}(F \log F)$
2: **for** $l \in 1, 2, 3, \ldots, L$ **do**
3: **for** $f \in \{2, 3, \ldots, \lceil F/(rl) \rceil\}$ **do**
4: $\omega_0 = 2\pi(f-1)/F$ ▷ $\mathcal{O}(1)$
5: $\boldsymbol{w}_l(\omega_0) = \boldsymbol{m}_l(\omega_0, \boldsymbol{f})$ ▷ $\mathcal{O}(1)$
6: Run Algorithm 2 line 16–27 recursively and obtain $\boldsymbol{a}_l(\omega_0)$ and $\boldsymbol{b}_l(\omega_0)$ that solve (46) and (47). ▷ $\mathcal{O}(l)$
7: $[\boldsymbol{J}]_{l,f} = \bar{\boldsymbol{w}}_l^T(\omega_0)\boldsymbol{a}_l(\omega_0) - \tilde{\boldsymbol{w}}_l^T(\omega_0)\boldsymbol{b}_l(\omega_0)$ ▷ $\mathcal{O}(l)$
8: **end for**
9: **end for**

6.1. Grid size selection

The accuracy of the methods are assessed via Monte Carlo simulations with R repetitions of three different experimental setups and compared with appropriate Cramér-Rao lower bounds (CRLB). We measure the root-mean-squared (RMS) value of the estimation accuracy. Some of the methods are possibly biased which motivates the use of the root-mean-squared metric. The additive noise is drawn from an *i.i.d.* Gaussian distribution $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}_N)$ so that the NLS estimator coincides with the ML estimator. The methods comprise the first two steps presented in the introduction, and the model order is assumed known. The refinement step is a Fibonacci line search and uses the exact NLS cost function for both the exact NLS and harmonic summation methods. We also evaluate the estimation accuracy for three different values of the number of grid points F, namely NL, $5NL$, and $20NL$ where F is related to the grid size $\Delta\omega_0$ derived in Sec. 4 via

$$F = \left\lceil \frac{2\pi}{\Delta\omega_0} \right\rceil . \tag{85}$$

First, we investigate the behaviour at a low fundamental frequency ω_0 uniformly distributed on $2\pi \cdot (0.75/N, 1.25/N)$. The amplitudes are constant $\sqrt{a_i^2 + b_i^2} = 1$, $i = 1, \ldots, L$ with the phase uniformly distributed on $(0, 2\pi)$. We do not use the YIN algorithm for problems involving a low fundamental frequency since, as we have previously

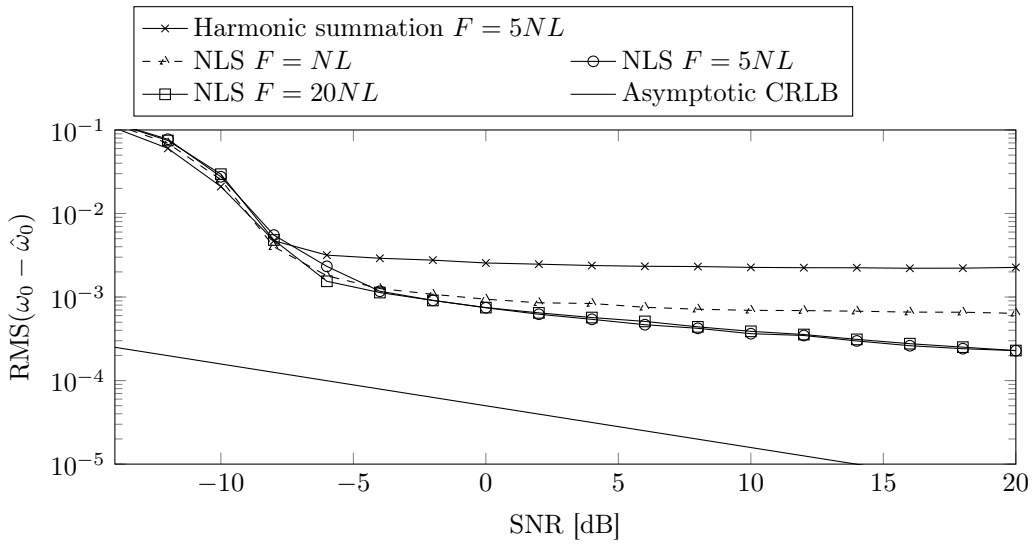

Figure 2: Estimator accuracy for different methods with $N = 500$, $L = 10$, $\omega_0 \in 2\pi[0.75/N, 1.25/N)$, $R = 10^4$ repetitions, constant amplitude $\sqrt{a_i^2 + b_i^2} = 1$, $i = 1, \ldots, L$ and phase drawn from a uniform distribution in the interval $(0, 2\pi)$.

discussed in the introduction, the autocorrelation methods do not work for such problems. Since the fundamental frequency and the phases are randomised between the Monte Carlo iterations, the exact CRLB also changes between the iterations. However, the asymptotic CRLB

$$\mathrm{var}(\hat{\omega}_0) \geq \frac{24\sigma^2}{N(N^2-1)\sum_{l=1}^{L}(a_l^2 + b_l^2)l^2} \tag{86}$$

is constant, and we therefore use that here instead although it is too optimistic when the fundamental frequency is low (see, e.g., [21]).

In Fig. 2, we observe the behaviour of the different methods as a function of the signal-to-noise-ratio (SNR). We observe that using $F = NL$ grid points is not dense enough and that at-least $F = 5NL$ grid points are needed for this setup. Furthermore, we note that the harmonic summation method is much worse than the other methods and has an "error-floor"-like behaviour, even though the exact cost-function is used in the refinement step (the second step described in the introduction). This indicates a selection of a non-true local maximum on the cost-function computed on the grid.

For higher frequencies, the methods can attain the asymptotic CRLB. To illustrate

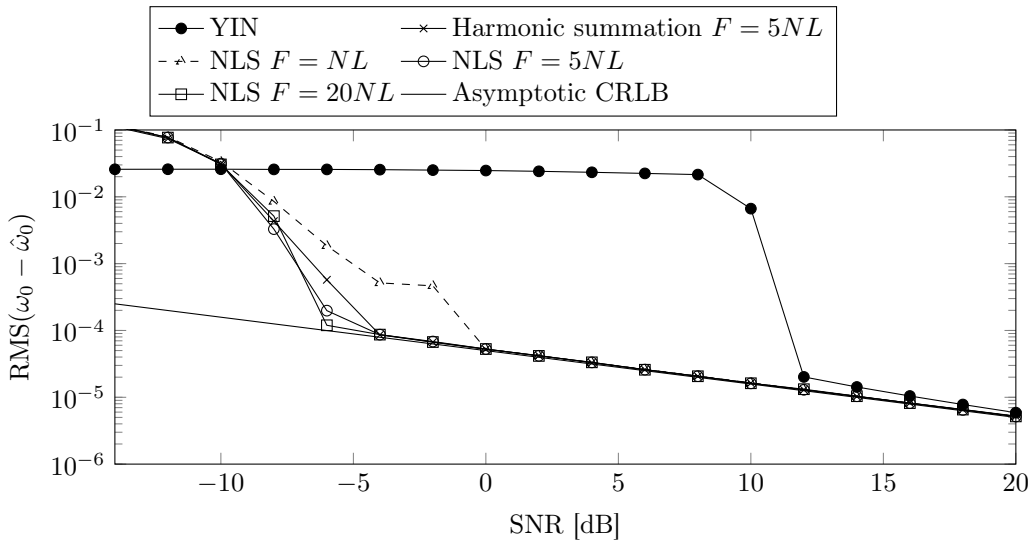

Figure 3: Estimator accuracy for different methods with $N = 500$, $L = 10$, $\omega_0 \in 2\pi[2/N, 4/N)$, $R = 10^4$ repetitions, constant amplitude $\sqrt{a_i^2 + b_i^2} = 1$, $i = 1, \ldots, L$ and phase drawn from a uniform distribution in the interval $(0, 2\pi)$.

this, we run the same setup as for Fig. 2, but with $\omega_0 \in 2\pi \cdot [2/N, 4/N)$ instead, and this produces the results shown in Fig. 3. Again, we see that the NLS method with a resolution of $F = NL$ is too low. We also see that both the NLS and the harmonic summation methods are optimal estimators for SNRs above -5 dB since they attain the asymptotic CRLB. The YIN algorithm, however, has a very bad estimation performance until around 12 dB. For SNRs above 12 dB, it still has suboptimal performance since it does not attain the CRLB.

To demonstrate that the NLS method is an asymptotically efficient estimator for even a low fundamental frequency, we also ran a simulation for fixed a_i, b_i, $i = 1, \ldots, L$ and ω_0. In Fig. 4, we show the results and the exact CRLB which has been calculated using the procedure described in [32, Sec. 3.9]. The results show that the NLS method attains the bound whereas the harmonic summation method has a poor performance for low fundamental frequencies. In this simulation, we get essentially the same performance for all grid sizes. However, since the fundamental frequency is not randomised between each run in this simulation, we cannot really draw any conclusions from this. Instead, we recommend that $F = 5NL$ uniform grid points are used which corresponds to $g \approx 1.15$

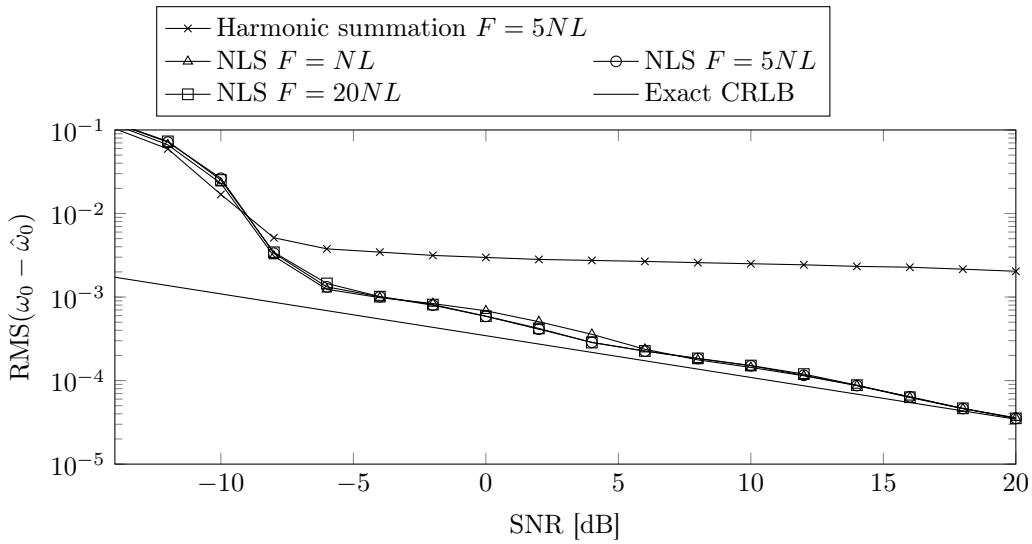

Figure 4: Estimator accuracy for different methods with $N = 500$, $L = 10$, $\omega_0 = \frac{3}{\pi}\frac{2\pi}{N}$, $R = 10^4$ repetitions for fixed a_i, b_i, $i = 1, \ldots, L$.

or that the maximum cost function value on the grid is at most 15 % smaller than the true maximum.

6.2. Computation time

To compare the computation speed of several different MATLAB implementations of the various algorithms, we ran a benchmark procedure. The timings were obtained by executing the algorithms 10^i times using the smallest $i \in \{0, 1, 2, \cdots\}$ such that the execution time $\tau_0 \geq 0.2\,\text{s}$. For this i, three repetitions of 10^i executions were then run and timed, producing the three repetition times τ_1, τ_2 and τ_3. The reported execution times in Fig. 5 and Fig. 6 were then calculated as $\tau = \min(\tau_1, \tau_2, \tau_3)/10^i$. All timings were executed on an Intel(R) Dual Core(TM) i5-2410M CPU at 2.3 GHz with Ubuntu Linux kernel 3.13.0-24-generic and MATLAB 8.4.0. Note that the computation time of MATLAB's FFT implementation can be quite high when the FFT size F is prime or has large prime factors. In an application where N and L are given, the FFT size and algorithm should be selected carefully so that this problem is avoided. Here, however, we do not handpick an FFT size and just use the value $F = 5NL$, corresponding to $g = 1.15$ in (85).

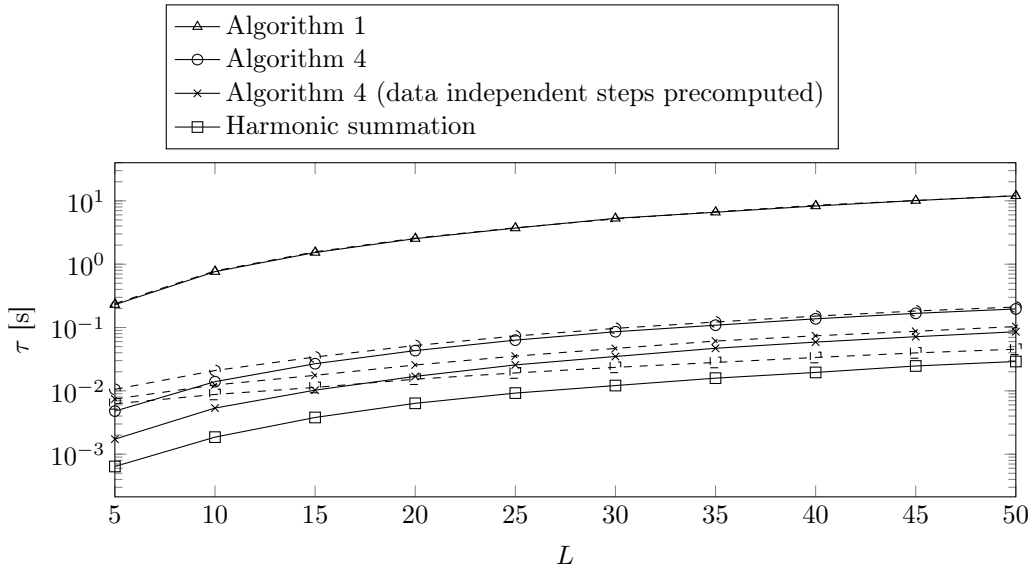

Figure 5: The computation time for four different ways of computing either the exact or the approximate NLS cost function for $N = 200$ and $F = 5NL$. The dashed lines indicate the computation time with the refinement step for all model orders.

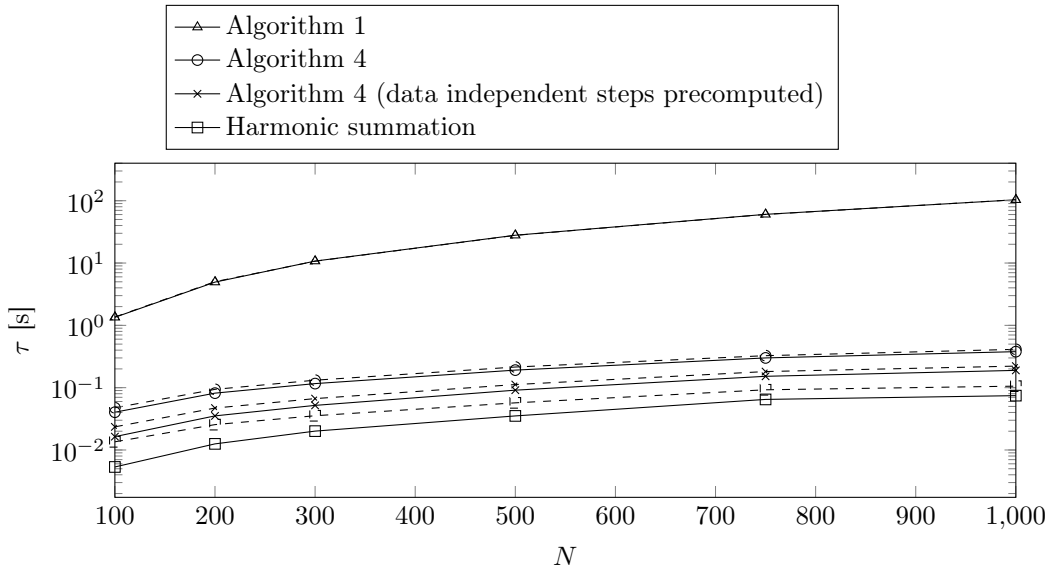

Figure 6: The computation time for four different ways of computing either the exact or the approximate NLS cost function for $L = 30$ and $F = 5NL$. The dashed lines indicate the computation time with the refinement step for all model orders.

In Fig. 5, we show the timings for $N = 200$ data points and a variable model order from $L = 5$ to $L = 50$. We see that the proposed algorithm is approximately 55–70 times faster compared to Algorithm 1. On the other hand the base line version of Algorithm 3 is approximately 7-9 times slower than harmonic summation in this implementation. This is to be expected since, even though harmonic summation and the proposed method asymptotically have the same time complexity, the constant for the latter algorithm is higher. It is also possible to pre-compute all data independent terms of Algorithm 3 and hence introduce a faster, but more memory costly method. This method is approximately 3–4 times slower than the harmonic summation method. In addition to showing the computation times for calculating the cost functions over a uniform grid for all model orders, the computation time with a refinement step for all model orders is also shown as dashed lines. Specifically, we have used a Fibonacci line search to refine the grid based estimate to within an accuracy of 10^{-7} rad/sample. For low model orders, the refinement step was more costly than the grid search for the fast NLS algorithms and the HS method. For the standard NLS algorithm, however, the cost of the refinement step was only a fraction of the total cost. In Fig. 6, the timings is reported for a fixed $L = 30$ and a variable number of data points from $N = 100$ to $N = 1000$. We observe a similar behaviour as in Fig. 5. However, the relative speed up of the proposed method in Algorithm 3 compared to Algorithm 1 increases with N and reaches a factor of approximately 150 at $N = 1000$.

7. Conclusion

In this paper, a computationally efficient method has been proposed for fundamental frequency estimation using the non-linear least-squares (NLS) method. The NLS method is statistically efficient if the noise is modelled as white and Gaussian and much more accurate than the autocorrelation based methods. Unfortunately, the NLS cost function has a very oscillating behaviour and hence the grid method is typically used as a reliable method to find the global maximum. We have shown how the grid size should be selected and that the evaluation of the cost function requires solving linear systems with a Toeplitz-plus-Hankel structure for the real-valued case. Moreover, we have analysed a recursive solver for Toeplitz-plus-Hankel systems and exploited this for computing the NLS

cost function for all model orders up to a certain specified maximum model order. The proposed algorithms have the same asymptotic time complexity as the harmonic summation method which is an approximate NLS method. The accuracy of the proposed method, the harmonic summation method and the YIN method was assessed in terms of Monte-Carlo studies. As expected the proposed method can attain the Cramér-Rao lower bound and is the most accurate method for low-fundamental frequencies. Timings of the MATLAB implementation show that the proposed method is of the same computational order as harmonic summation, although with a larger constant. This is in agreement with the conducted algorithmic analysis.

References

[1] N. H. Fletcher, T. D. Rossing, The Physics of Musical Instruments, 2nd Edition, New York, NJ, USA: Springer-Verlag, 1998.

[2] M. G. Christensen, A. Jakobsson, Multi-Pitch Estimation, San Rafael, CA, USA: Morgan & Claypool, 2009.

[3] H. Dudley, The carrier nature of speech, Bell System Technical Journal 19 (4) (1940) 495–515.

[4] R. J. Sluijter, The development of speech coding and the first standard coder for public mobile telephony, Ph.D. thesis, Techniche Universiteit Eindhoven (2005).

[5] G. L. Ogden, L. M. Zurk, M. E. Jones, M. E. Peterson, Extraction of small boat harmonic signatures from passive sonar, J. Acoust. Soc. Am. 129 (6) (2011) 3768–3776.

[6] S. Gade, H. Herlufsen, H. Konstantin-Hansen, N. J. Wismer, Order tracking analysis, Technical review no. 2, Brüel & Kjær A/S (1995).

[7] V. K. Murthy, L. J. Haywood, J. Richardson, R. Kalaba, S. Salzberg, G. Harvey, D. Vereeke, Analysis of power spectral densities of electrocardiograms, Mathematical Biosciences 12 (1–2) (1971) 41–51.

[8] L. R. Rabiner, On the use of autocorrelation analysis for pitch detection, IEEE Trans. Acoust., Speech, Signal Process. 25 (1) (1977) 24–33.

[9] W. Hess, Pitch Determination of Speech Signals: Algorithms and Devices, Berlin Heidelberg, Germany: Springer-Verlag, 1983.

[10] D. Talkin, A robust algorithm for pitch tracking (RAPT), in: W. B. Kleijn, K. K. Paliwal (Eds.), Speech Coding and Synthesis, Vol. 495, Elsevier, 1995, Ch. 14, pp. 495–518.

[11] A. de Cheveigné, H. Kawahara, YIN, a fundamental frequency estimator for speech and music, J. Acoust. Soc. Am. 111 (4) (2002) 1917–1930.

[12] H. Kawahara, A. de Cheveigné, H. Banno, T. Takahashi, T. Irino, Nearly defect-free F0 trajectory extraction for expressive speech modifications based on STRAIGHT, in: Proc. Interspeech, 2005, pp. 537–540.

[13] S. Gonzalez, M. Brookes, PEFAC - a pitch estimation algorithm robust to high levels of noise, IEEE Trans. Audio, Speech, Lang. Process. 22 (2) (2014) 518–530.

[14] J. K. Nielsen, T. L. Jensen, J. R. Jensen, M. G. Christensen, S. H. Jensen, Fast and statistically efficient fundamental frequency estimation, in: Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., 2016, pp. 86–90.

[15] R. O. Schmidt, Multiple emitter location and signal parameter estimation, IEEE Trans. Antennas Propag. 34 (3) (1986) 276–280.

[16] R. Roy, T. Kailath, ESPRIT-estimation of signal parameters via rotational invariance techniques, IEEE Trans. Acoust., Speech, Signal Process. 37 (7) (1989) 984–995. doi:10.1109/29.32276.

[17] B. G. Quinn, P. J. Thomson, Estimating the frequency of a periodic function, Biometrika 78 (1) (1991) 65–74. arXiv:http://biomet.oxfordjournals.org/content/78/1/65.full.pdf+html.

[18] J. R. Jensen, G.-O. Glentis, M. G. Christensen, A. Jakobsson, S. H. Jensen, Fast LCMV-based methods for fundamental frequency estimation, IEEE Trans. Signal Process. 61 (12) (2013) 3159–3172.

[19] A. M. Noll, Pitch determination of human speech by the harmonic product spectrum, the harmonic sum spectrum, and a maximum likelihood estimate, in: Proc. of the symposium on computer process. commun., Vol. 779, 1969.

[20] D. J. Hermes, Measurement of pitch by subharmonic summation, J. Acoust. Soc. Am. 83 (1) (1988) 257–264.

[21] M. G. Christensen, Accurate estimation of low fundamental frequencies from real-valued measurements, IEEE Trans. Audio, Speech, Lang. Process. 21 (10) (2013) 2042–2056.

[22] H. Li, P. Stoica, J. Li, Computationally efficient parameter estimation for harmonic sinusoidal signals, Signal Processing 80 (9) (2000) 1937 – 1944.

[23] J. Tabrikian, S. Dubnov, Y. Dickalov, Maximum a-posteriori probability pitch tracking in noisy environments using harmonic model, IEEE Trans. Speech Audio Process. 12 (1) (2004) 76–87.

[24] J. K. Nielsen, M. G. Christensen, S. H. Jensen, Default Bayesian estimation of the fundamental frequency, IEEE Trans. Audio, Speech, Lang. Process. 21 (3) (2013) 598–610.

[25] J. K. Nielsen, M. G. Christensen, A. T. Cemgil, S. H. Jensen, Bayesian model comparison with the g-prior, IEEE Trans. Signal Process. 62 (1) (2014) 225–238.

[26] R. Wu, J. Li, Z.-S. Liu, Super resolution time delay estimation via MODE-WRELAX, IEEE Trans. Aerosp. Electron. Syst. 35 (1) (1999) 294–307.

[27] J. K. Nielsen, T. L. Jensen, J. R. Jensen, M. G. Christensen, S. H. Jensen, Grid size selection for nonlinear least-squares optimisation in spectral estimation and array processing, in: Proc. European Signal Processing Conf., 2016, pp. 1653–1657.

[28] I. Gohberg, I. Koltracht, Efficient algorithm for Toeplitz plus Hankel matrices, Integral Equations Operator Theory 12 (1) (1989) 136–142.

[29] G. Heinig, K. Rost, Fast algorithms for Toeplitz and Hankel matrices, Linear Algebra and its Applications 435 (1) (2011) 1–59.

[30] T. Kailath, A. H. Sayed, Displacement structure: Theory and applications, SIAM Review 37 (3)

(1995) pp. 297–386.

[31] G. Merchant, T. Parks, Efficient solution of a Toeplitz-plus-Hankel coefficient matrix system of equations, IEEE Trans. Audio, Speech, Lang. Process. 30 (1) (1982) 40–44.

[32] S. M. Kay, Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory, Englewood Cliffs, NJ, USA: Prentice Hall PTR, 1993.